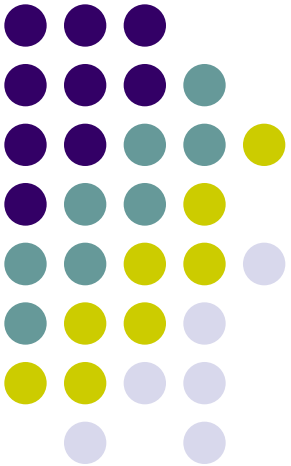
Технологии программирования

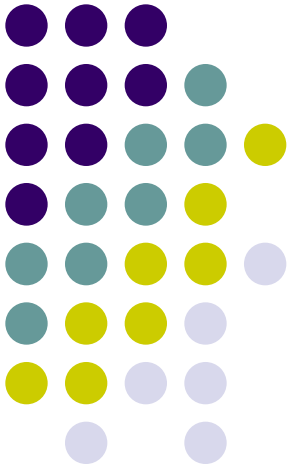
Программирование на языке Java

Газейкина Анна Ивановна, канд.пед. наук, доцент кафедры информатики, информационных технологий и методики обучения информатике УрГПУ

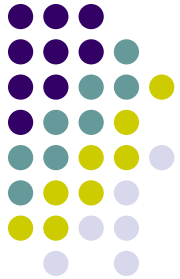


Введение в программирование на Java

Лекция 1

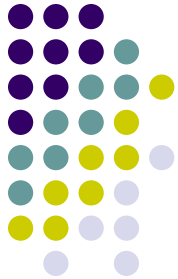


Немного истории



- Язык Java разработан в 90-е годы прошлого века в компании Sun Microsystems (в 2009 г. была поглощена американской компанией Oracle Corporation).
- Автором Java считается канадец Джеймс Гослинг.

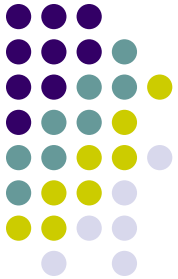
Java Development Kit



Основные инструменты, необходимые для создания программ на языке Java, включены в **JDK** – это комплект разработчика приложений на языке Java.

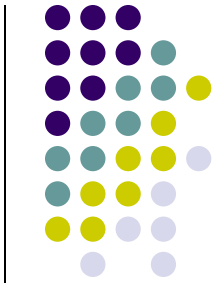
JDK - программное обеспечением с открытым исходным кодом, его можно скачать свободно и бесплатно с официального сайта Oracle Corporation (**www.oracle.com)**.

Состав JDK:



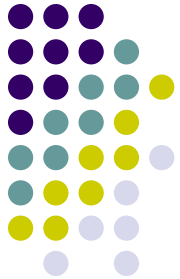
- компилятор Java (**javac**);
- документация;
- примеры;
- утилиты;
- исполнительная система Java (Java Runtime Environment – **JRE**).

Состав Java Runtime Environment:



- виртуальная java-машина (интерпретатор **java**);
- библиотеки классов Java.

Редакции JDK:



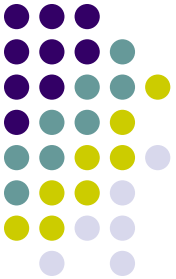
- **Standart Edition (SE)** – для создания и исполнения приложений для индивидуального использования (или использования в масштабах малого предприятия);
- **Enterprise Edition (EE)** – для создания коммерческих приложений для крупных и средних предприятий;
- **Micro Edition (ME)** – для разработки приложений для устройств, ограниченных в ресурсах (сотовых телефонов, карманных персональных компьютеров и т.п.).

Интегрированные среды разработки приложений на Java

(Integrated Development Environment – **IDE**) :

- NetBeans IDE,
- Sun Java Studio Creator,
- IntelliJ IDEA,
- Borland JBuilder,
- **Eclipse.**

Состав IDE:

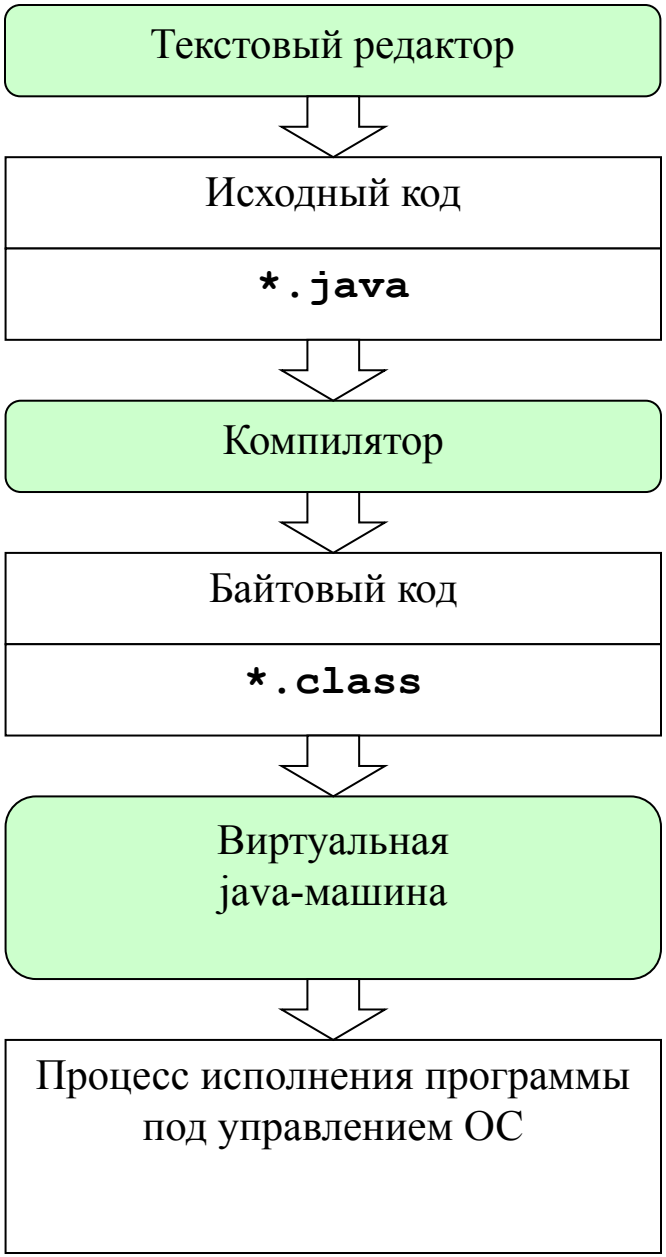


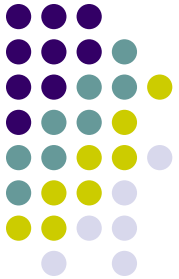
- специализированный **текстовый редактор**, который облегчает форматирование текста программы, подсвечивает синтаксис и предоставляет прочие удобства;
- **отладчик** – программный комплекс для поиска и исправления ошибок в программе;
- **фоновый компилятор**, который указывает на синтаксические ошибки еще в процессе набора текста программы;
- **справочная система**.

Основная особенность Java

КРОССПЛАТФОРМЕННОСТЬ (!):

- Windows
- Linux
- Mac OS
- Solaris
- Android
- и другие ОС.

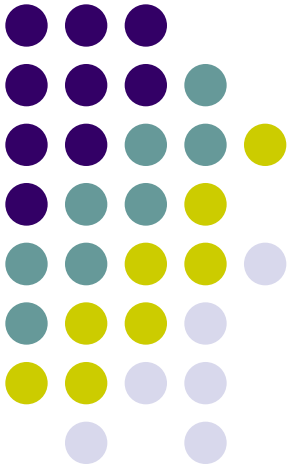

Текстовый редактор
Исходный код
*.java
Компилятор
Байтовый код
*.class
Виртуальная
java-машина
Процесс исполнения программы
под управлением ОС



- Для исполнения программы на компьютере должна быть установлена JRE, java-машина интерпретирует байтовый код, учитывая особенности данной ОС (исполняемый файл НЕ создается).
- НО: Исполняемый файл (нативный бинарный код для конкретной операционной системы) создать можно при помощи сторонних программ-утилит.

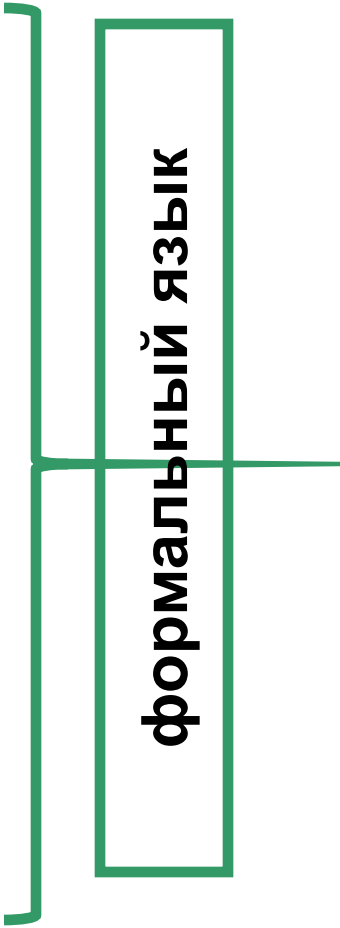
Структура Java-программы. Линейные программы

Лекция 2

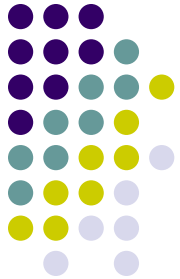


Java как формальный язык

- **алфавит** – конечное упорядоченное множество символов;
- **синтаксис** – правила записи конструкций (операторов) из символов алфавита;
- **семантика** – смысловая сторона языковых конструкций;
- **прагматика** – последствия практического применения языка.

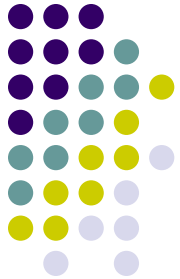


Алфавит языка Java



- зарезервированные слова;
- символы для записи идентификаторов:
  - буквы (в том числе и национальных алфавитов, например, русского, но все же использовать нежелательно);
  - десятичные цифры;
  - символ _ (подчеркивание);
- разделители;
- специальные символы.

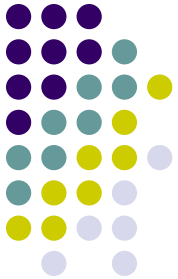
Имя (идентификатор)



- это последовательность из букв, цифр и символа _ (подчеркивание), начинающаяся не с цифры.
- Поименован должен быть каждый объект программы для того, чтобы к нему можно было обратиться.

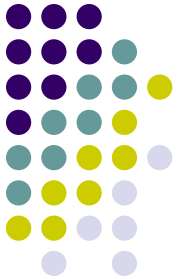
Объявление (описание) любого объекта должно предшествовать вызову (обращению).

Структура Java-программы



- **Программа** – это класс (Java – это объектный язык).
- **Класс** – это совокупность полей и методов.
- **Поля** – это данные (переменные).
- **Методы** реализуют алгоритмы обработки данных (это подпрограммы: процедуры и функции).

Структура Java-программы



```
public class Name //заголовок
{
// тело программы
}
```

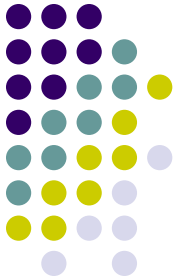
ВАЖНО:

Компилятор Java ЧУВСТВИТЕЛЕН к регистру:

a≠A

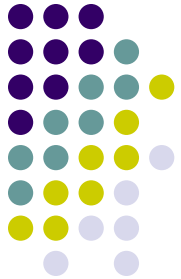
Имена классов принято записывать с ЗАГЛАВНОЙ БУКВЫ.

Составной оператор (блок)



```
{ оператор1;
  оператор2;
  …
  операторN; }
```

Переменная



- это ячейка памяти для хранения единицы данных.

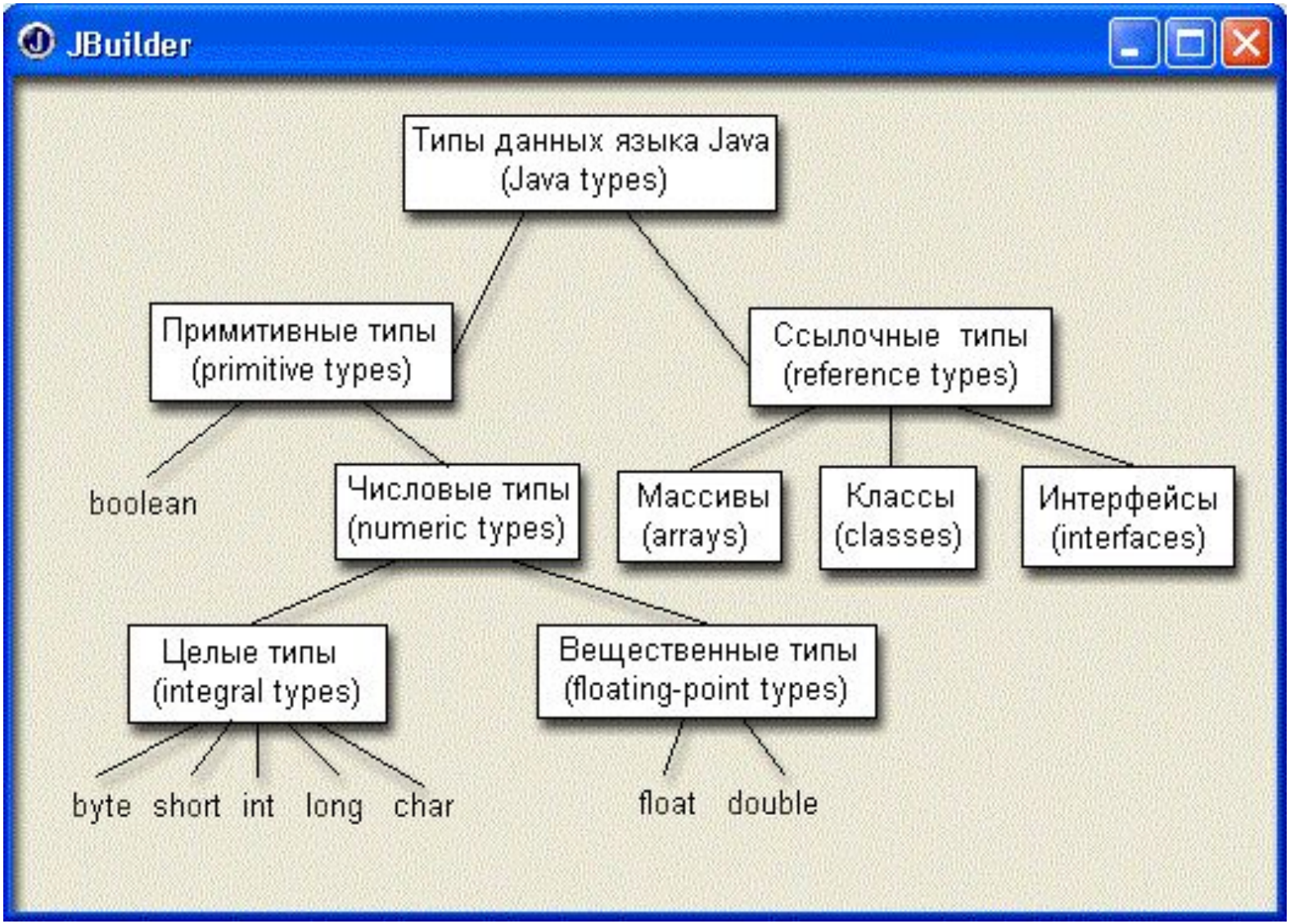
Характеристики переменной:

- имя (идентификатор);
- тип *(Java – язык со строгой типизацией данных)*;
- значение.

Тип данных определяет:

- диапазон значений данных (как данные представляются в ОП, какой объем памяти требуется для их хранения);
- операции (действия), которые можно над этими данными выполнять.

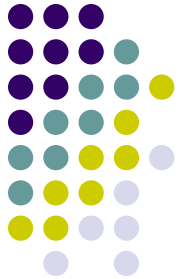
Типы данных Java



Примитивные типы данных

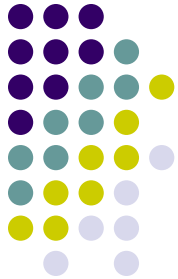
тип	*объем памяти*	*диапазон значений*
byte	8 бит	-128..127
short	16 бит	-32 768..32 767
int	32 бита	-2 147 483 648 .. 2 147 483 647
long	64 бита	64 разрядное целое
float	32 бита	3.4e-038 .... 3.4e+ 038
double	64 бита	1.7e-308 .... 1.7e+ 308
char	16 бит	символ в кодировке Unicode
boolean	8 бит	false, true

Операции над данными:



- унарные:
  - изменение знака числа **-**, отрицание **!**
- бинарные:
  - мультипликативные:
    - умножение *****, деление **/**, остаток от деления нацело **%**, конъюнкция **&**
  - аддитивные:
    - сложение **+**, вычитание **-**, дизъюнкция **|**
  - операции отношения:
    - **==**, **!=**, **<**, **>**, **<=**, **>=**

Объявление переменной

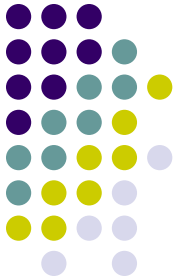


позволяет задать ИМЯ и ТИП переменной, может быть выполнено в любом месте программы:

тип имя;

```
int a;
double x, y;
char c1, c2;
boolean f;
```

Оператор присваивания



позволяет задать ЗНАЧЕНИЕ переменной

имя = выражение;

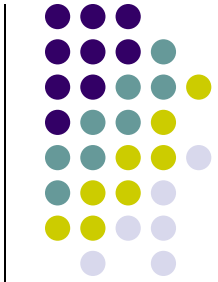
знак присваивания

выполняется СПРАВА налево:

- сначала вычисляется значение выражения;
- затем это значение записывается в переменную с указанным именем.

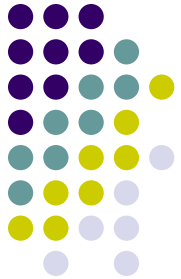
! **ТИПЫ значения выражения и переменной должны СОВПАДАТЬ.**

Виды оператора присваивания



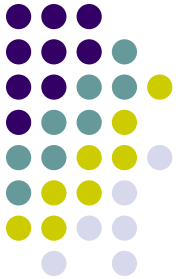
- традиционный:
  - a = 10;
  - b = a+8;
  - b = b+1;
- совмещенный с объявлением переменной (инициализация):
  - int k = 0;
  - double x = 3.5, pi = 3.14159;

Виды оператора присваивания



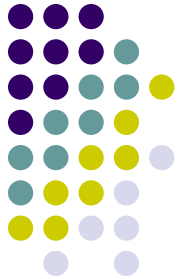
- инкремент (увеличение значения целочисленной переменной на 1):
  - k++; // постфиксная форма
  - ++k; // префиксная форма
- декремент (уменьшение значения целочисленной переменной на 1):
  - k--; // постфиксная форма
  - --k; // префиксная форма

Виды оператора присваивания



- запись выражения в левой части:
  - k + = 5; // k = k + 5;
  - n * =2; // n = n * 2;

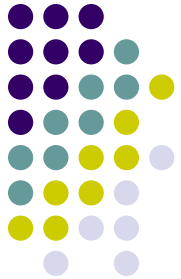
Вывод информации в консоль



СПРАВКА: Консо́ль компьютера (англ. *console* - пульт управления) - это совокупность устройств (в том числе устройств ввода-вывода), обеспечивающая взаимодействие человека-оператора с компьютером.

Консоль - также разновидность текстового интерфейса (в противоположность графическому).

Вывод информации в консоль

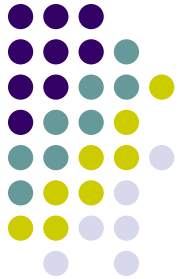


используются методы **print()** и **println()**, примененные к объекту **out** класса **PrintStream**, созданному в общедоступном классе **System**:

System.out.print (*строка***);** *//выводит строку*

System.out.println (*строка***);** *//выводит строку и символ перевода строки*

Вывод информации в консоль – примеры:



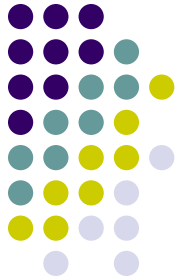
System.out.print(“Всем привет”);

System.out.println(“В “+n+” сутках ” + (n*24) + “ часов ”);

Числовые данные (и данные других примитивных типов) **приводятся** к строковому типу (**String**) **автоматически** (автоматически вызывается метод **toString()**).

!

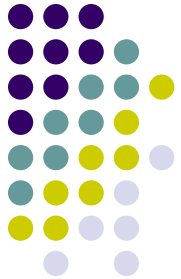
Методы



(детально будут рассмотрены в более поздних лекциях).

Методы реализуют алгоритмы обработки данных (это подпрограммы: процедуры и функции).

Метод main

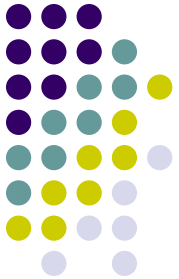


- должен присутствовать в классе-программе;
- является ТОЧКОЙ ВХОДА в программу – с него начинается исполнение;
- имеет описание:

  public static void main (String [ ] args) {

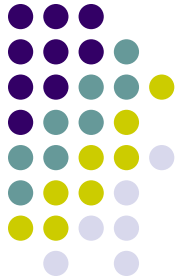
  …}

Пример простейшей программы



```
public class Hello {
    public static void main (String [ ] args) {
     System.out.println (“Всем привет!!!”);
    }
}
```

Пример другой простейшей программы



```
public class Simple {
    public static void main (String [ ] args) {
     int n=10;
     System.out.println (“В памяти переменная n имеет
    значение ” + n);
     n *=2;
     System.out.println (“А теперь переменная n имеет
    значение ” + n);
    }
}
```

Важное правило

Символ **;** (**точка с запятой**) ставится ПОСЛЕ любого ОПЕРАТОРА.

Этот символ воспринимается компилятором как ПУСТОЙ ОПЕРАТОР (оператор, который «ничего не делает»).